Course Title: Advanced Cryptography
Course code: ICT-6115

Submitted By:

Md. Tarikul Islam

IT-23616

Submitted To:

Mr.Ziaur Rahman

Associate Professor

Dept. of ICT, MBSTU.

IT-23616

① Implications of quantum computing on cryptography -

Quantum computers pose a serious threat to traditional cryptosystems like RSA and Ecc because of shor's algorithm. This algorithm can efficiently solve the problems on which RSA and Ecc rely, like factoring large numbers and solving the discrete logarithm problem. This means quantum computers could break these crypto-systems, making them insecure.

The implication includes :-

1. Loss of confidentiality
2. compromised integrity
3. Long-term security risk.

IT-23616

Post-Quantum cryptographic Algorithms:

To secure communications in the quantum era, researchers are developing post quantum cryptography. These are cryptographic algorithms belived to be secure even against quantum computers.

1. Lattice-based cryptography.
2. code-based cryptography:

   Based on hand decoding problems which quantum computers can't speed up.
3. multivariate polynomial cryptography:

   Based on solving system of polynomials which quantum computers struggle with.
4. Hash-based cryptography: uses hash functions, which are resistant to quantum attacks.
5. Isogeny-based cryptography:

   Based on the difficulty of finding isogenies between elliptic curves.

IT-23616

which is secure against quantum attacks.

How These Algorithm Resist Quantum Attacks –

↳ Lattice based: Hard for quantum computers to solve problems like LWE and SVP.

↳ code-based: Quantum computers can't speed up decoding random codes.

↳ multivariate: Quantum computers can't efficiently solve systems of polynomials.

↳ Hash-based: Quantum computer can't break hash functions easily.

↳ Isogeny based: Quantum algorithms can't solve the isogeny problem efficiently.

(2) Here's simple implementation of PRNG in Python using the current timestamp, Process ID (os.pid) and a modulus operation to constrain the output within a given range:

Python code:

```
import os
import time

class SimplePRNG:

    def __init__(self, range_min, range_max):
        self.range_min = range_min
        self.range_max = range_max

    def generate(self):
        # Get current time in milliseconds.
        timestamp = int(time.time() * 1000)

        pid = os.getpid()

        random_value = (timestamp * pid + 12345) %
                       (self.range_max - self.range_min
                       +1) + self.rang_min .
        return random_value

# Example usage
prng = SimplePRNG(1, 100)
for _ in range(10):
    print(prng.generate()) .
```

IT-23616

(3) Comparison between Traditional ciphers and modern Symmetric ciphers;

Feature	Traditional ciphers	modern symmetric ciphers
Key length	Short (1-26 for caesar cipher).	Long (56 bit for DES and 128/192/256 bit for AES.
Encryption speed	Fast	Fast but computationally heavier.
Decryption speed	Same as encryption	similar to encryption optimize for speed.
Security	weak against brute force, frequency analysis and pattern. detection.	Strong against brute force and statistical attack.
Vulnerability to crypto analysis	Highly vulnerable.	Resistant to linear and differential cryptaanalysis.

IT-23616

(4) Let S be the set of all 2-element subsets of $\{1,2,3,4\}$.

$S = \{\{1,2\}, \{1,3\}, \{1,4\}, \{2,3\}, \{2,4\}, \{3,4\}\}$

Each permutation $\alpha \in S_4$ acts on a 2 element subset $\{a,b\}$ by permuting its elements:

$$\alpha.\{a,b\} = \{\alpha(a), \alpha(b)\}$$

Proving the action is well defined.

The result $\{\alpha(a), \alpha(b)\}$ is always a 2-element subset of $\{1,2,3,4\}$.

Since S_4 is a group, the action respects the group operation.

Thus, the action is well-defined.

IT-23616

orbit and stabilizer of {1,2}

orbit of {1,2}:

The orbit consists of all 2-element subsets of {1,2,3,4}, as any permutation of S_4 can map {1,2} to any other 2-element subset.

The orbit contains 6 subsets, so the size of the orbit is 6.

Stabilizer of {1,2}:

The stabilizer is the subgroup of S_4 that keeps {1,2} unchanged. This includes the identity permutation and the permutation that swaps 1 and 2.

Therefore, the stabilizer contains 2 elements. So its size is 2.

IT-23616

(5)

(i) Show that GF(2^2) forms a group under multiplication.

The elements of GF(2^2) are $\{0, 1, \alpha, \alpha+1\}$.

$\alpha \times \alpha = \alpha^2 = \alpha + 1$

Since $\alpha^2 + \alpha + 1 = 0$.

So, the set is closed under multiplication

Multiplication in GF(2^2) is associative.

The identity element for multiplication is 1, because for any element a in GF(2^2), we have $a \times 1 = a$.

Every non-zero element in GF(2^2) hay a mutiplicative inverse:

↳ 1 is its own inverse.

↳ $\alpha \times (\alpha+1) = 1$, so α's inverse is $\alpha+1$

↳ $(\alpha+1) \times \alpha = 1$, so $\alpha+1$'s iverse is α.

Since all foun properties are satisfied GF(2^2) forms a group under multiplication.

(5) (ii) Verify whether the set of all nonzero elements of $GF(2^2)$ is cyclic.

To verify whether the group of nonzero elements of $GF(2^2)$ is cyclic, if there exists an element $g \in \{1, \alpha, \alpha+1\}$ such that the power of g generate all the nonzero elements of $GF(2^2)$.

We will check the powers of each elements in $\{1, \alpha, \alpha+1\}$.

powers of 1:

$1^1 = 1$ and no other element is generated.

Power of α:

$\alpha^1 = \alpha$,

$\alpha^2 = \alpha + 1$

$\alpha^3 = \alpha.(\alpha+1) = 1$.

So, the power of α generate $\{\alpha, \alpha+1, 1\}$ which are all the nonzero elements of $GF(2^2)$.

Since α generates all the nonzero elemens of $GF(2^2)$, the set of nonzero elements of $GF(2^2)$ is cyclic, and α is a generator of the group.

IT-23616

⑥ Let $GL(2,R)$ be the general linear group of 2×2 invertible matrices over R. Show that the set of scalar matrices forms a normal subgroup of $GL(2,R)$. Construct the corresponding factor group, and interpret its structure.

Scalar matrices in $GL(2,R)$

A scalar matrix is a matrix of the form:

$$\lambda I = \begin{pmatrix} \lambda & 0 \\ 0 & \lambda \end{pmatrix}$$

where $\lambda \in R \setminus \{0\}$ and I is the identity matrix.

The set of scalar matrices forms a subgroup of $GL(2,R)$ because:

↳ The product of two scalar matrices is a scalar matrix.

↳ The identity matrix I is a scalar matrix.

↳ Every scalar matrix has an inverse.

Thus, the set of scalar matrices is a subgroup of $GL(2,R)$.

The Factor Group $GL(2,R)/S$, where S is the set of scalar matrices, can be thought of as the group of all matrices in $GL(2,R)$ where scalar matrices are identified as equivalent. This group is known as the projective general linear

IT23616

group; $PGL(2,R) = GL(2,R)/S$.

Interpretation of $PGL(2,R)$

The group $PGL(2,R)$ describes the symmetries of the real projective plane. It is the group of transformations of the projection plane that are equivalent to invertible 2×2 matrices, but where scalar multiples of a matrix are considered equivalent.

Thus, $GL(2,R)/S \cong PGL(2;R)$, which is a key group in geometry and the study of projective transformation.

IT-23616

(7)

Diffie-Hellman Key Exchange Protocol

The Diffie-Hellman key exchange protocol is a cryptographic method that allows two parties to securely exchange a shared secret over an insecure communication channel.

Diffie-Hellman allows two parties who have not previously exchanged any keys agree on a secret key. Alice and Bob agree on a prime modulus p and a primitive element g.

Alice picks a random number x and sends $a = g^x \bmod p$ to Bob.

Similarly, Bob picks a random number y and sends $b = g^y \bmod p$ to Alice.

Alice then computes $b^x \bmod p = g^{xy} \bmod p$ and Bob computes $a^y \bmod p = g^{xy} \bmod p$.

The computed value $g^{xy} \bmod p$ is then used as a secret key. An attacker overhearing the conversation between Alice and Bob can not learn $g^{xy} \bmod p$.

IT-23616

Application in secure communication

After establishing a shared secret, symmetric encryption can be used for encrypting the data exchanged between the parties.

Security Against Attacks

1. man-in-the-middle Attack (MITM)

In a man-in-the-middle-attack, an attacker intercepts and alters the communication between Alice and Bob. In the Diffie-Hellman protocol, this attack would involve the attacker intercepting A and B and replacing them with their own values.

2. Discrete Logarithm Problem

Given three integers a, b and m. Find an integer k such that $a^k \equiv b \pmod m$ where a and m are relatively prime. If it is not possible for any k to satisfy this relation, print -1.

IT-23616

Impact of Insufficiently Large Prime modulus -

If the prime modulus P used in Diffie-Hellman is not sufficiently large, the protocol's security is compromised:

- Brute-force Attack: A small P makes the discrete logarithm problem easier to solve because there are fewer possible values for the private keys. Attackers could use brute-force methods to calculated the private key from the public key.
- Efficiency of Attacks: modern cryptographic attacks such as Polland's Rho or Index calculus, are more efficient for smaller prime numbers, further reducing the security of the protocol.

(8) Proof: Let G be a group, and let H_1 and H_2 be subgroups of G. We want to show that their intersection $H_1 \cap H_2$ is also a subgroup of G.

Since H_1 and H_2 are both subgroups they contain the identity elements e of G. So, e is in $H_1 \cap H_2$.

If $x \in H_1 \cap H_2$, then x is in both H_1 and H_2. Since both are subgroups, x^{-1} is in both H_1 and H_2, so $x^{-1} \in H_1 \cap H_2$.

Since the intersection contains the identity and is closed under inverses, $H_1 \cap H_2$ is a subgroup of G.

Example:

Let $G = Z_6$ (integers modulo 6 under addition) and consider:

$H_1 = \{0, 2, 4\}$ (subgroup of Z_6)

$H_2 = \{0, 3\}$ (subgroup of Z_6)

The intersection $H_1 \cap H_2 = \{0\}$, which is the trivial subgroup of Z_6.

Thus, the intersection of two subgroups, is a subgroup

IT-23616

(c) Proof that Z_n is commutative;

The ring Z_n is the set of integeny modulo n, with addition and multiplication defined modulo n.

(i) commutatively of Addition:

Addition in Z_n is commutative because $(a+b) \bmod n = (b+a) \bmod n$ for all $a, b \in Z_n$.

(ii) commutatively of multiplication:

multiplication in Z_n is commutative because $(a \cdot b) \bmod n = (b \cdot a) \bmod n$ for all $a, b \in Z_n$.

Since both addition and multiplication are commutative; Z_n is commutative.

Zero Divisons in Z_n:

A zero divisor in a ring is an element a such that $a \cdot b = 0$ for some nonzero b.

In Z_n, zero divisors exist if n is not a prime number. This is because for composite n, there exist nonzero elements a and b such that $a \cdot b \equiv 0 \bmod n$.

For example, in Z_6,

$2 \cdot 3 = 6 \equiv 0 \bmod 6$,

So 2 and 3 are zero divisons.

IT-23616

For Z_n to be a field, every nonzero element must have a multiplicative inverse, which occurs if and only if n is prime.

↳ When n is prime: Z_n is a field because every nonzero element has an inverse modulo n.

↳ When n is composite: Z_n is not a field because there will be zero divisors (elements without inverses).

IT-23616

(10) Vulnerabilities of DES:

The Data Encryption Standard (DES), once a widely used encryption algorithm, is now consider insecure for modern use due to several vulnerabilities:

(i) Key length:

· DES uses a 56-bit key for encryption. This key size is relatively small by today's standard.

(ii) Brute-Force Attacks:

Brute-force attacks are the most straightforward way to break DES. since there are only 2^{56} possible keys, it is feasible for an attacker to try every possible key until the correct one is found.

(iii) Weaknesses in the algorithm:

DES was designed in the 1970s and while it was considered secure at the time, it was found to be vulnerable to cryptanalytic attacks such as differential and linear cryptanalytic

IT-23616

Impact of Key Length on Security:

The key length is a fundamental aspect of the security of any encryption algorithm. In DES, the 56-bit key means that there are 2^{56} possible keys.

① Exponential Growth:

The number of possible keys grows exponentially with the key length. For example, a 128-bit key would have 2^{128} possible keys which is far more difficult to brute force than 2^{56}. This is why modern encryption standard use much larger keys (128 bit or 256-bit) to maintain strong security.

② DES's Small Key Space:

The 56-bit key of DES way shown to be too small. With modern GPUs and specialized hardware, the key space can be searched very quickly. This vulnerability in key length makes DES unsuitable for securing information today.

IT-23616

Development of AES and its improvements:

The advanced Encryption Standard (AES) was developed to replace DES, addressing its key limitations, especially in terms of key size and cryptographic resistance.

Key size:

AES supports key sizes of 128 bits, 192 bits and 256 bits, making it significantly more secure than DES with its 56-bit key. The longer key sizes mean that the number of possible keys is vastly increased, making brute-force attacks infeasible.

It uses a much more complex structure and a different design compared to DES, making it more robust against these type of attacks.

AES is not only more secure but also more efficient than DES. It is faster in both software and hardware implementations and is widely used in modern cryptographic applications.

IT-23616

(11)

(i) The Feistel structure of DES helps resist differential cryptanalysis in several ways:

(1) Division into two halves:
The Feistel structure splits the data block into two halves (left and right) and in each round, one half of the data is modified using a function that depends on the other half.

(ii) Non-linear function:
DES uses a non-linear function in each round, which makes the relationship between input and output more complex and harder for attackers to predict.

(iii) multiple rounds: The algorithm repeats the Feistel process over 16 rounds. increasing confusion and making it more difficult to attack the differences through the rounds.

(iv) key mixing: Each round uses a different part of key, which further complicates attacks and prevents attackers from easily determining the key.

IT-23616

(11)

(ii) Advanced Encryption Standard is more resistant to differential cryptanalysis compared to DES due to the following reasons:

1. SubBytes: This step involves a non linear substitution using a fixed S-Box. The S-box is designed to have good resistance against differential cryptanalysis by making the relationship between input and output highly complex. In contrast, DES uses simpler S box, which are more vulnerable to such attacks.

2. ShiftRow: In AES, the ShiftRows operation involves rotating the rows of the state in a cyclic manner. It makes it more difficult to track the changes through the cipher over multiple rounds.

3. mixcolumns: The mixcolumns operation in AES ensures that each byte in a column of the state is mixed with others

providing better diffusion. This operation increases the complexity of predicting how differences in the plaintext will propagate through the ciphertext. In DES, the equivalent operation, the Feistel function, does not offer as much diffusion, making it more vulnerable.

(4) AddRoundKey: AES uses the AddRoundKey operation, where each round's key is XORed with the state ensuring key mixing and complicating the relationship between input and output. While DES also mixes the key in each round, AES's more complex structure particularly with multiple operations (SubBytes, ShiftRows, MixColumns).

AES has a more robust structure with better diffusion and confusion, due to its multiple rounds of complex operation,

IT-23616

(13)

① ECB (Electronic codebook) mode works by encrypting each block of plaintext independently using the same key. If the same plaintext block appears multiple times in the data, ECB will produce the same ciphertext block for each identical plaintext block.

Let's say the plaintext is made up of repeated blocks $P_1, P_2 \ldots, P_k$ and each P_i is identical.

For a block cipher with block size n, the encryption function E applied to each block P will give us:

$$c_i = E(K, P_i)$$

where K is the encryption key and c_i is the ciphertext block corresponding to P_i.

Since $P_i = P_j$ for all i, j, we have

$$c_i = c_j \text{ for all } i, j.$$

This means that if two plaintext blocks are the same, their corresponding ciphertext blocks will also be identical.

(13)
(11) If an error occurs in one ciphertext block during decryption, it will affect two plaintext blocks:

1. Immediate error: An error in c_i will cause the incorrect decryption of P_i because $P_i = D(K, c_i) \oplus c_{i-1}$, so the erroneous ciphertext c_i will result in an incorrect plaintext P_i'.
2. Chained error: since the plaintext block P_{i+1} depends on c_i (i.e. $P_{i+1} = D(K, c_{i+1}) \oplus c_i$) the error in c_i will propagate and also affect P_{i+1}.

Limited Error propagation:-

The key observation is that the error caused by a corrupted ciphertext block only affects the corresponding plaintext and the next plaintext block.

↳ A single bit error in c_i affects P_i immediately and the error propagates to P_{i+1} but it does not affect any further blocks beyond P_{i+1}.

↳ Thus, error propagation is limited to just two blocks (one affected block and the next one).

IT-23616

(14)

Vulnerability to Known-Plaintext Attacks:-

1. Linearity of LFSRs:

The sequence generated by an LFSR is a deterministic and periodic binary sequence. If the attacker knows a few output bits of the LFSR, they can exploit the linear relationship between these bits to deduce the internal state of the LFSR.

The linearity makes this extraction feasible because the next bit in the sequence depends linearly on the previous ones.

2. Known-Plaintext Attack:

In a known-plaintext attack, an attacker has access to both the ciphertext and the corresponding plaintext. The ciphertext is typically generated by XORing the plaintext with the key-stream. If the attacker knows enough of the key-stream bits, they can calculate the key stream using the known relation $c_i = P_i \oplus K_i$, where c_i is the ciphertext, P_i is the plaintext and K_i is the key-stream bit.

Mathematical method to mitigate this vulnerability:
To mitigate the vulnerability of LFSRs to known-plaintext attacks, one approach is to introduce non-linearity into the key-stream generation process. Non-linear functions increase the complexity of predicting the future bits of the sequence and make it significantly harder to deduce the internal state of the LFSR.
For example, using XOR or AND operations between the outputs of two or more LFSRs can make the key stream non-linear and harder to predict, even if some key-stream bits are known. This significantly increases the complexity of the attack.

In simple terms, by making the key-stream generation non-linear, it becomes much harder to deduce the internal state of the LFSRs, thus protecting the system from known-plaintext attacks.

IT-23616

(A4)

Vulnerability to Known-Plaintext Attacks:-

1. Linearity of LFSRs:

The sequence generated by an LFSR is a deterministic and periodic binary sequence. If the attacker knows a few output bits of the LFSR, they can exploit the linear relationship between these bits to deduce the internal state of the LFSR.

The linearity makes this extraction feasible because the next bit in the sequence depends linearly on the previous ones.

2. Known-Plaintext Attack:

In a known-plaintext attack, an attacker has access to both the ciphertext and the corresponding plaintext. The ciphertext is typically generated by XORing the plaintext with the key-stream. If the attacker knows enough of the key-stream bits, they can calculate the key stream using the known relation $c_i = P_i \oplus K_i$, where c_i is the ciphertext, P_i is the plaintext and K_i is the key-stream bit.

IT-23616

(15)

(i) Shannon's Definition of perfect secrecy mathematically:

Shannon's definition of perfect secrecy is based on the condition that, given the ciphertext, the probability distribution of the plaintext should remain unchanged. In other words, the ciphertext should give no additional information about the plaintext.

mathematically, perfect secrecy is defined as:

$P(M = m/c = c) = P(M = m)$ for all $m \in M$ and $c \in C$.

where:

m is the set of plaintexts

c is the set of ciphertexts

$m \in M$ represents a specific plaintext massage.

$c \in C$ represents a specific ciphertext ~~massage~~.

$P(m = m/c = c)$ is the conditional probability that the plaintext is m given the ciphertext is c.

$P(m = m)$ is the prior probability that the plaintext is m.

IT-23616

(1) Proving that the One-Time Pad Achieves Perfect Secrecy:

The One-Time Pad (OTP) is a Symmetric encryption scheme where the key is randomly chosen, and the key is as long as the plaintext. The encryption and decryption processes are as follows:

Encryption: $c = m \oplus k$

Decryption: $m = c \oplus k$.

To prove perfect secrecy, we need to show that for all possible plaintexts m and ciphertexts c, the probability $P(M = m/c = c)$ is equal to $P(M = m)$. meaning that the ciphertext c does not provide any information about the plaintext m, which satisfies the condition for secrecy.

IT-23626

⑮ (iii) critically Analyzing why perfect Secrecy is impractical for Large-Scale communication systems:

It is impractical for large-scale communication systems for several reasons:-

① Key management:

The one-Time Pad requires the key to be as long as the plaintext and to be used only once. This means that for each communication, a new, randomly generated key of equal length must be securely exchanged and stored.

2. Key Reuse and Storage;

Reusing keys in the one-Time Pad would break its perfect secrecy, so every message requires a fresh key.

3. Scalability:

In a large-scale communication system, the number of messages and users grows exponentially.

④ Storage Requirements:

The key must be stored securely on both the senden's and receiver's ends, increasing the storage burden on both parties. In a large system, this would mean maintaining vast amounts of key data.

⑤ Vulnerability to key compromise:

If any key is compromised, the security of all messages encrypted with that key is lost.

IT-23616

(16)

Let's compute the first 5 numbers of the LCG sequence using the given recurrence relation:

$$x_{n+1} = (ax_n + c) \bmod m$$

Let's use the following values:

$a = 5$ (multiplier)

$c = 1$ (increment)

$m = 16$ (modulus)

Initial seed $x_0 = 7$.

With these values, the recurrence relation becomes:

$$x_{n+1} = (5x_n + 1) \bmod 16$$

Now, let's compute the first 5 numbers of the sequence starting with $x_0 = 7$.

1. For x_1:

$$x_1 = (5 \times 7 + 1) \bmod 16$$
$$= 36 \bmod 16$$
$$= 4$$

2. For x_2:

$$x_2 = (5 \times 4 + 1) \bmod 16$$
$$= 21 \bmod 16$$
$$= 5$$

IT 23616

(3) For x_3:

$x_3 = (5 \times 5 + 1) \bmod 16$

$= 26 \bmod 16$

$= 10$

4. For x_4:

$x_4 = (5 \times 10 + 1) \bmod 16$

$= 51 \bmod 16$

$= 3$

5. For x_5:

$x_5 = (5 \times 3 + 1) \bmod 16$

$= 16 \bmod 16$

$= 0$

The first 5 numbers of the LCG sequence are:

$x_0 = 7,\ x_1 = 4,\ x_2 = 5,\ x_3 = 10,\ x_4 = 3,\ x_5 = 0.$

IT-23616

(17) **Ring:** A ring is a set with two operations, addition and multiplication, that follow certain rules. Rings are algebric structures that generalize fields. These operations must satisfy a few key properties. Formally, a ring R is defined by the following:

1. Additive closure: For all $a, b \in R$, $a+b \in R$.
2. Additive Associativity: For all $a, b, c \in R$ $(a+b)+b = a+(b+c)$.
3. Additive Identity: There exists an element $0 \in R$ such that $a+0 = a$ for all $a \in R$.
4. Additive Inverses: For each element $a \in R$, there exists an element $-a \in R$ such that $a+(-a) = 0$.
5. multiplicative closure: For all $a, b \in R$, $a \cdot b \in R$.
6. multiplicative Associativety: For all $a, b, c \in R$ $(a \cdot b) \cdot c = a \cdot (b \cdot c)$.
7. Distributivity of multiplication over Addition: For all $a, b, c \in R$, the multiplication distributes over addition:
   $a \cdot (b+c) = (a \cdot b) + (a \cdot c)$
   $(a+b) \cdot c = (a \cdot c) + (b \cdot c)$.

IT-2361 6

Examples of Rings:

Commutative Ring: The set of integers $\mathbb{Z}$ (with the usual addition and multiplication) is a commutative ring.

↳ Addition: $a+b$ is always an integer.

↳ multiplication: $a.b$ is always an integer.

Non-commutative Ring:

The set of $n \times n$ matrices over the real numbers $M_n(R)$ (with matrix addition and multiplication) is a non-commutative ring.

matrix multiplication is generally not commutative, meaning $AB \neq BA$ for some matrices A and B.

IT- 2361G

Rings and Finite Fields:

A finite field (also known as a Galois field) is a field that contains a finite number of elements. A field is a special type of ring where every non-zero element has a multiplicative inverse and multiplication is commutative.

Role of Rings in cryptographic algorithms like RSA:

In cryptography, rings and fields play crucial roles, especially in algorithms like RSA.

① RSA and modulo operations:

RSA encryption works in the ring Z_n, where n is the product of two large prime numbers. The structure of the ring Z_n enables modular arithmetic to be performed efficiently. operations such as modular exponentiation and modular inversion are central to RSA.

In RSA, key generation involves selecting two primes P and q, computing $n = P \times q$

and using the properties of the ring Z_n to find public and private keys.

mathematical operations in RSA:

The encryption and decryption processes in RSA use modular exponentiation, which is based on the properties of the ring Z_n. Specifically these operations involve raising numbers to certain exponents modulo n, which is a ring operation.

The security of RSA depends on the difficulty of factoring large numbers, a problem that is related to the structure of rings and their factorization properties.

IT-23616

(18)

(1) Given,

$P = 5, q = 11$

compute $n = P \cdot q = 5 \cdot 11 = 55$

$\phi(n) = (P-1).(q-1) = 4 \cdot 10 = 40$

choose $e = 3$ (where $\gcd(3, 40) = 1$)

compute d such that

$e \cdot d = 1 \pmod{40}$ and $d < 40$

value is $d = 27$ Since $3 \times 27 = 81 = 2 \times 40 + 1$

The public key is $(e, n) = (3, 55)$

The private key is $(d, n) = (27, 55)$

Let the message $m = 2$

Encryption:

ciphertext $c = m^e \bmod n$

$= 2^3 \bmod 55 = 8 \bmod 55$

ciphertext, $c = 8$

Decryption:

using the decryption formula:

$m = c^d \bmod n$

$= 8^{27} \bmod 55$

message, $m = 2$

IT-23616

(18) (ii)

Given, $p=7$, $q=3$

calculate g: $g = h^{(p-1)/q} \bmod p$

choose $h = 2$

$g = 2^{(7-1)/3} \bmod 7$.

$= 2^2 \bmod 7$

$= 4$

Private key: $x = 5$

calculate public key:

$y = g^x \bmod p = 4^5 \bmod 7 = 1024 \bmod 7 = 2$

Keys:

Public key: $\{p, q, g, y\} = \{7, 3, 4, 2\}$

Private key: $\{p, q, g, x\} = \{7, 3, 4, 5\}$

Signature Generation:

Given, $H(m) = 3$

Random integer: $k = 2$

calculate r: $r = (g^k \bmod p) \bmod q$

$= (4^2 \bmod 7) \bmod 3$

$= (16 \bmod 7) \bmod 3$

$= 2 \bmod 3$

$r = 2$

calculate s: $s = k^{-1}(H(m) + x \cdot r) \bmod q$

$k^{-1} = 2^{-1} \bmod 3 = 2$

$s = 2(3 + 5 \cdot 2) \bmod 3 = 26 \bmod 3 = 2$

Signature: $\{r, s\} = \{2, 2\}$.

IT-23616

Verification:

Given, message digest: $H(m) = 3$

compute w: $w = s^{-1} \bmod q$

$= 2^{-1} \bmod 3$

$= 2$

compute u_1: $u_1 = H(m) \cdot w \bmod q$

$= 3 \cdot 2 \bmod 3$

$= 6 \bmod 3$

$= 0$

compute u_2: $u_2 = r \cdot w \bmod q$

$= 2 \cdot 2 \bmod 3$

$= 4 \bmod 3$

$= 1$

verify v:

$v = (g^{u_1} \cdot y^{u_2} \bmod p) \bmod q$

$= (4^0 \cdot 2^1 \bmod 7) \bmod 3$

$= (1 \cdot 2 \bmod 7) \bmod 3$

$= (2 \bmod 7) \bmod 3$

$= 2 \bmod 3$

$= 2$

Since $v = r$, the signature is valid.

IT-23616

19

(1) Given the elliptic curve equation is:

$$y^2 = x^3 + ax + b \pmod p$$

where, $p = 23$

$a = 1$

Substituting $a = 1, b = 1, p = 23$ and $b = 1$

the equation becomes:

$$y^2 \equiv x^3 + x + 1$$

For the point $P = (3, 10)$:

$$y^2 = 10^2 = 100$$

Now compute the right-hand side of the equation

$$x^3 + x + 1 = 3^3 + 3 + 1$$

$$= 27 + 3 + 1$$

$$= 31$$

Now ~~we have~~ reduce modulo 23:

$31 \bmod 23 = 8$

Now we have:

$$100 \equiv 8 \pmod{23}$$

Now, reduce $100 \bmod 23$:

$100 \div 23 = 4$ (quotient), $100 - 4 \times 23 = 100 - 92 = 8$

Thus, $100 \equiv 8 \pmod{23}$.

Therefore, the point $P = (3, 10)$ lies on the elliptic curve, since $y^2 \equiv x^3 + x + 1 \bmod 23$.

IT-23616

(22) (i) Key characteristics of a secure Hash Function:

(I) Pre-image Resistance

(II) Second Pre-image Resistance

(III) collision Resistance

(IV) Deterministic

(V) Fixed Length

(VI) Efficient.

(ii) Impact of output Hash Length:

A longer hash makes it harder to find two inputs with the same hash.

It increases security because we need more computational power to break it.

Ex. finding a collision in SHA-256 requires 2^{128} attempts.

IT-23616

(iii) Real-world uses of SHA:

Digital Signatures: SHA helps ensure the integrity and authenticity of messages. It hashes the message, which is then signed by a private key.

Blockchain: In blockchain, SHA-256 is used to secure transactions and link blocks in the chain.

Data Integrity: SHA checks if data is altered by comparing hashes before and after transfer.

Password Hashing: Instead of storing passwords, systems store the hash of the password to keep it secure.

IT-23616

(23) (i).

The Shortest vector problem (SVP) is a fundamental problem in lattice-based cryptography. It involves finding the shortest non-zero vector in a lattice, where a lattice is a grid-like structure in multidimensional space.

Role in security: The difficulty of solving SVP is what makes lattice based cryptographic schemes secure. In these ~~ch~~ schemes, breaking the encryption relies on solving SVP, which is considered hard, even for quantum computers. This provides resistance against attacks that might break traditional schemes like RSA or ECC using Shor's algorithm.

IT-23616

(23) (11)

Lattice-based cryptography: The security of lattice-based cryptography relies on hard problems like SVP and Learning with Errors (LWE). These problems are belived to be resistant to quantum attacks because quantum computers struggle to solve them efficiently.

RSA and Ecc: RSA and Ecc are based on the difficulty of factoring large numbers (RSA) or solving the elliptic curve discrete logarithm problem. Shor's algorithm can solve both of these problems efficiently using quantum computers, making RSA and Ecc insecure in the post quantum era.

Lattice-based cryptography is designed to be secure against quantum attacks, while RSA and Ecc are vulnerable to Shor's algorithm.

IT-23616

(23) (iii) Quantum cryptography: Quantum cryptography focuses on using the principles of quantum mechanics, like superposition and entanglement, to create secure communication channels. Its most famous application is Quantum Key Distribution (QKD), which allows two parties to securely share a key, even in the presence of an eavesdropper.

Lattice based cryptography: Lattice based cryptography, on the other hand, relies on mathematical problems in lattices (like SVP and LWE) that are hard to solve, even for quantum computers.

Quantum cryptography uses quantum mechanics to secure data, while lattice based cryptography uses hard mathematical problems to ensure security, making it more suited for protecting against quantum computer attacks in a classical setting.

IT-23616

(24)

LFSR and Polynomial: The LFSR's behavior is described by a characteristic polynomial P(x). If this polynomial is primitive, it ensures that the LFSR goes through all possible non-zero states before repeating.

Primitive Polynomial: A primitive polynomial of degree m over GF(2) generates a maximal length sequence, meaning the LFSR cycles through all non-zero states. Since there are 2^m possible states and the all-zero state is not used, the LFSR can go through $2^m - 1$ different states.

maximum period: Because a primitive polynomial guarantees the LFSR will generate all non-zero states before repeating, the maximum period of the keystream is $2^m - 1$.

(25) (i) signing a message using an LWE-based signature scheme

1. key generation:
   - ↳ Private key: choose a secret vector s_k.
   - ↳ public key: Generate matrix A and vector b such that $b = A \cdot s_k + e$ where e is a small error.

2. Signing a message:
   - ↳ convert the message m into a vector m.
   - ↳ choose a random vector r.
   - ↳ compute $v = A \cdot r + m \bmod q$.
   - ↳ The signature is $\sigma = (r, v)$

3. verification:

   check if $A \cdot r + m = v \bmod q$.

   If true, the signature is valid.

IT-23616

(25)

(11) Signing a message using Public and Private keys

1. message: convert the message m into a vector m.
2. Random vector: choose a random vector r.
3. Signature: compute $v = A \cdot r + m \bmod q$
   & the signature is $a = (r, v)$
4. Security: The LWE problem ensures it's hard for an attacker to forge the signature without the private key s_k.
5. Verification: verify $A \cdot r + m \equiv v \bmod q$

The LWE problem ensures that it is difficult to forge a signature without private key because solving the underlying LWE problem is hard.